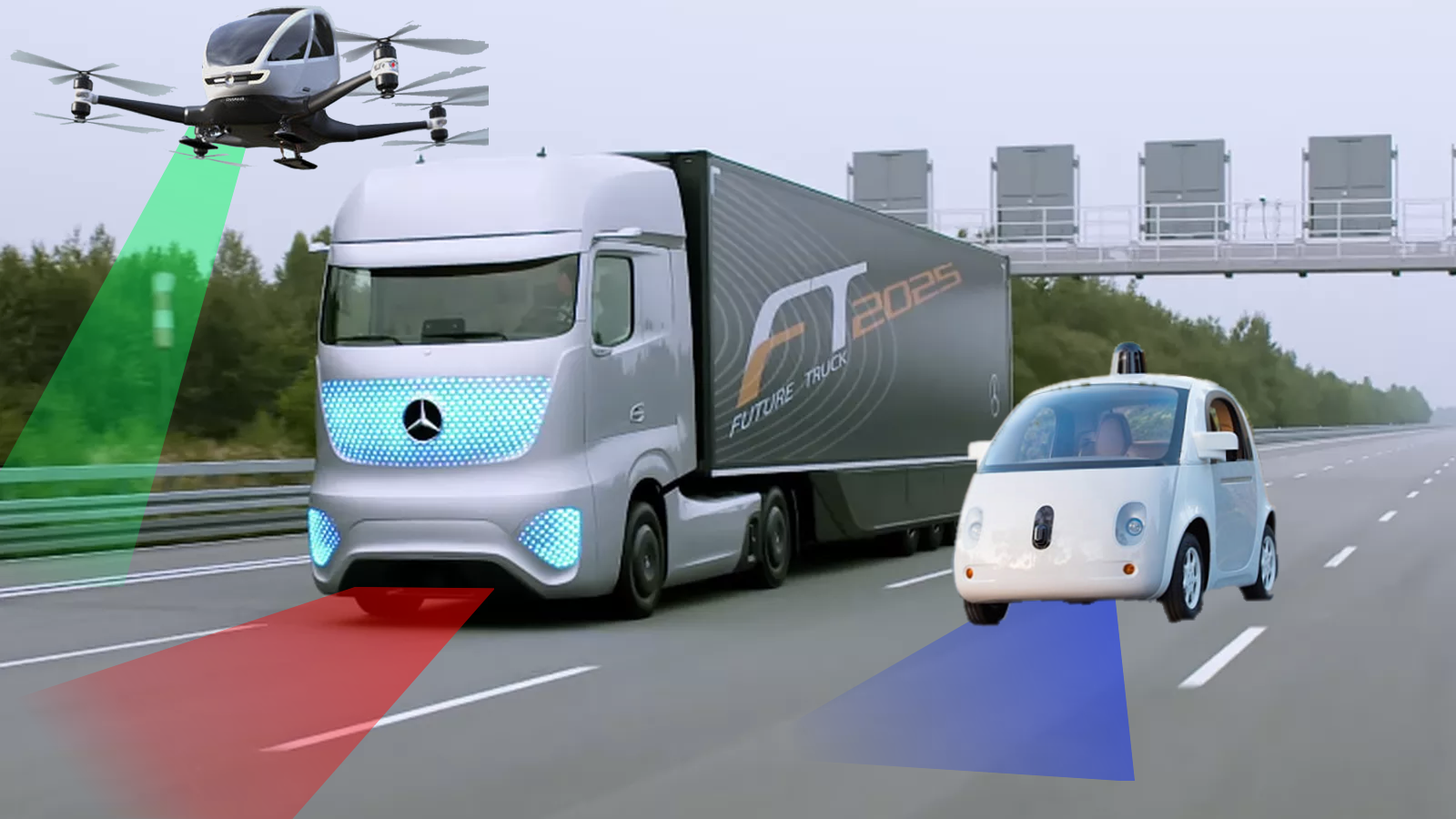
**DESARROLLO BASADO EN AGENTES**

**CURSO 2016-2017**

**GUIÓN DE LA TERCERA PRÁCTICA**

**“Gugel Vehicles” para búsqueda distribuida**



28 de Noviembre de 2016

Luis Castillo Vidal

[L.Castillo@decsai.ugr.es](mailto:L.Castillo@decsai.ugr.es)

**Objetivo de la práctica**

Esta práctica consiste en la implementación de un equipo de agentes heterogéneos, con capacidades diferentes, en la que se va a poner especial énfasis en la capacidad de auto-organización del equipo de agentes para distribuir sus objetivos y en su capacidad de gestionar recursos compartidos de forma que colaboren en un objetivo común.

**Contenido**

[Entrega de material](#_dbg586p8f0fg)

[Fecha](#_ac05niodoodx)

[Material](#_jpdnj1t2pndh)

[Descripción de la práctica](#_k00umnrccfe9)

[Arquitectura de la práctica](#_6mupbyn2qird)

[El entorno del agente](#_fq1aokhy6bz8)

[La implementación de la solución](#_a6trymaq8p42)

[Operativa de la práctica](#_gr3ispisn6e1)

[Conexión a la plataforma](#_g3yx2botsf2q)

[Los agentes del servidor](#_cg79irnxkws5)

[Protocolo de comunicación con los agentes del servidor](#_52plyfddi8d9)

[Uso de recursos compartidos](#_l0kngkrilq4n)

[Seguimiento de la práctica](#_nevzq68augg)

[Material necesario para la práctica](#_eg6vuhfvd7or)

[Ficheros relacionados](#_4hmihrcx8oiw)

[Argumentos motivadores](#_ghbtc6qkv2r0)

[Mundos posibles para conectarse](#_ldzsiwk9n4qf)

[Evaluación de la práctica](#_e0a280tkbq2)

[Anexo I. Diagrama de secuencia](#_sacdsoqrfxxz)

[Anexo II: Shenron el resucitador](#_b7nui4ypml4o)

# Entrega de material

### Fecha

Siguiendo el [calendario oficial de la asignatura](https://docs.google.com/spreadsheets/d/106hgmCYA85GYT4WVS8Rr6uERLniPmbDNaiJ7jFAO_Rk/edit#gid=0) será el **18 de Enero de 2017** de 9:30 a 11:30 durante la sesión de defensa del trabajo en grupo.

### Material

Se deberá depositar en la carpeta compartida del grupo la siguiente documentación y siguiendo las normas que se indican.

1. Código fuente.
   1. Todas las clases desarrolladas pertenecerán al mismo paquete “practica3”.
   2. Cada clase/método deberá estar correctamente documentado en [JavaDoc](http://es.wikipedia.org/wiki/Javadoc) **indicando claramente el autor/autores** que han contribuido en cada sección.
   3. Un fichero zip con todos los ficheros .java utilizados así como las bibliotecas no estándar utilizadas (excepto las de Magentix).
2. Documentación
   1. Un texto con un máximo de 7 páginas donde se comenten los modelos de agente utilizados y los aspectos más importantes del desarrollo y sobre todo las consideraciones más importantes sobre la **sociedad de agentes** diseñada así como una tabla con los resultados obtenidos por cada mundo resuelto (ver [mundos disponibles](#7v7xjtua7mej))
      1. Nombre del mundo
      2. Mejor sesión
         1. Conversation-ID de la sesión
         2. Número y tipo de agentes inicial
         3. Número y tipo de agentes que han llegado a la meta
   2. Una presentación de 7 minutos en la que el equipo exponga las características más importantes de la práctica.

# Descripción de la práctica

El problema que se deberá resolver es el **diseño e implementación** de un sistema multiagente, inspirado en la práctica anterior, que emula un problema de búsqueda distribuida en una ciudad. Los vehículos empleados deberán localizar un punto objetivo, cuya posición se desconoce, pero que se puede percibir con el mismo radar de la Práctica 2, en caso de estar suficientemente cerca. Hay que tener en cuenta que cada vehículo podrá encontrar obstáculos en el camino que deberá bordear para no chocar con ellos. Estos obstáculos pueden ser muros dentro del mapa, el borde del mapa o bien otros vehículos presentes en la misma posición. Igualmente los vehículos deberán organizarse para utilizar un recurso compartido vital: las recargas de la batería son limitadas y podrían llegar a agotarse antes de que todos los vehículos lleguen al objetivo.

### Arquitectura de la práctica

La arquitectura del desarrollo de la práctica estará distribuida entre múltiples plataformas tal y como se indica en la figura, estando la parte del servidor (en color morado) ya desarrollada (controla el entorno del agente) y disponible para su uso y los alumnos deberán centrarse en construir (al menos) los cuatro agentes externos (vehículos de búsqueda) los cuales podrán ejecutarse desde los laboratorios de prácticas o desde un ordenador portátil.



#### El entorno del agente

El entorno del agente está completamente controlado por los agentes del servidor y puede percibirse de distintas formas que cada grupo deberá configurar para un mejor aprovechamiento del agente vehículo. Todos los vehículos deberán intentar alcanzar la zona donde se encuentra el objetivo sin chocar con los obstáculos.

### La implementación de la solución

Se deberán implementar los vehículos y todos aquellos objetos y agentes adicionales que se consideren necesarios como agentes de la plataforma Magentix2 con el objetivo de llevar al máximo número de vehículos hasta el objetivo. En esta práctica la auto-organización entre los agentes es muy importante para acceder a los recursos compartidos (espacio y energía) de forma ordenada. Para facilitar el proceso de búsqueda, los mundos de ejemplo son “sin caminos sin salida”, es decir, una vez llegado a un punto no será necesario deshacer el camino para poder continuar.

# Operativa de la práctica

### Conexión a la plataforma

Se hará utilizando la conocida API de Magentix

AgentsConnection.connect(host,port,virtualhost,username, password, ssl);

Para intentar garantizar la privacidad del trabajo de cada grupo y evitar interferencias entre los grupos, se proveerán datos de conexión diferentes y privados para cada grupo, los cuales se recomienda mantener en privado.

### Los agentes del servidor

En la plataforma hay, al menos, un agente <controlador>, el cual es el principal interlocutor con el agente vehículo para entrar al mundo virtual y moverse por él. El verdadero nombre del agente controlador es diferente para cada grupo y se encuentra conectado a un virtualhost distinto y deberá mantenerse de forma privada.

Este mismo agente <controlador> percibe todo el movimiento, el mundo virtual y la percepción del estado de cada uno de los cuatro vehículos de forma síncrona con cada actividad que realicen éstos. Todos los agentes disponen exactamente de los siguientes sensores.

1. Radar. Rastrea un área de celdas alrededor del vehículo y devuelve el estado del mundo en ese área indicado con un número entero
   1. 0 = celda libre
   2. 1 = obstáculo
   3. 2 = borde del mundo
   4. 3 = el objetivo
   5. 4 = cualquier vehículo del mismo equipo
2. Monitor de batería del vehículo. Desde el estado inicial de la batería del vehículo cada movimiento realizado consume batería. Este sensor informa sobre el estado de la misma. Si este sensor mide un nivel de batería por debajo del 1% se entenderá que el vehículo se ha estrellado en ese punto.
3. GPS. Aunque la dimensión del mundo es desconocida, y la conexión del vehículo a cada mundo le hace aparecer en un punto diferente del mismo, la posición del vehículo se puede representar como un par de coordenadas discretas (X,Y) que ocupa dentro del mundo. Es necesario tener en cuenta que el origen de coordenadas es la esquina superior izquierda.

### Protocolo de comunicación con los agentes del servidor

El [diagrama de secuencia del Anexo I](#ywbbevw1g0w2) ilustra el diálogo con el agente controlador, en el que se pueden observar las siguientes operaciones.

1. **SUBSCRIPCIÓN** 
   1. SUBSCRIBE {“world”:”<m>”}
   2. Es la primera operación que se debe realizar para entrar en un mundo virtual y debe indicar en el JSon del contenido del mensaje el mundo en el que se quiere loguear (según los nombre de la [sección correspondiente](#7v7xjtua7mej)).
   3. Esta operación puede realizarla cualquiera de los vehículos implementados por el grupo, pero en cuanto se reciba en el agente controlador, se iniciará una nueva sesión de trabajo para todos los vehículos, con una clave de acceso nueva.
   4. Si el registro es correcto, el controlador responde con un CONVERSATION-ID sólo al vehículo que ha solicitado la subscripción, debiendo encargarse éste vehículo de compartirla con el resto de vehículos y de incorporarla a cualquier mensaje que se intercambie con el servidor durante esta sesión de trabajo.
2. **REGISTRO**
   1. REQUEST {“command”:”checkin”} :CONVERSATION-ID @
   2. Esta operación debe hacerla cada vehículo por separado una única vez por cada sesión y siempre después de haber realizado la subscripción. Implica registrarse en el mundo virtual tras lo cual el vehículo recibirá la siguiente información:
      1. El OK del registro
      2. Una descripción de sus capacidades, pues cada agente podría ser diferente a los demás:
         1. “fuelrate”, un número entero que describe cuántas unidades de fuel consume en cada movimiento. Los valores permitidos son 1, 2 y 4.
         2. “range”, un número entero que describe la amplitud de la percepción con su radar. Si se recibe “fuelrate”:5 significa que el radar percibe un área de 5x5 celdas alrededor del vehículo. Los valores permitidos son 3, 5, y 11.
         3. “fly”, un booleano que describe si el vehículo puede volar o no, es decir, puede atravesar los obstáculos que detecte en el radar (radar == 1) aunque no puede atravesar los bordes del mundo (radar == 2). Los valores permitidos son true y false.
      3. El CONVERSATION-ID de la sesión de trabajo que está abierta
      4. Un campo REPLY-WITH con un ID que el vehículo deberá devolver en el siguiente mensaje que envíe al controlador dentro de un campo IN-REPLY-TO (al igual que se hizo en el hackathon)
   3. En el momento del registro, si es correcto, se le asigna al vehículo un punto de despegue aleatorio, situado en el borde inferior o superior del mundo virtual, seleccionado también de forma aleatoria.
3. **MOVIMIENTO**
   1. REQUEST {“command”:”moveX”} :CONVERSATION-ID @ :IN-REPLY-TO #
   2. Tiene las mismas consideraciones que el GugelCar de la Práctica 2. Las razones por las que se puede recibir un “CRASHED” (debidamente informadas por el controlador con una performativa “FAILURE”) es que el vehículo ha agotado su batería (“BATTERY EXHAUSTED”), se ha chocado con un borde (“CRASHED ON BORDER”) o con un obstáculo (“CRASHED ON WALL”) o ha chocado con otro vehículo (“CRASHED WITH AGENT”).
4. **REPOSTAJE**
   1. REQUEST {“command”:”refuel”} :CONVERSATION-ID @ :IN-REPLY-TO #
   2. Tiene las mismas consideraciones que el GugelCar de la Práctica 2. La única excepción es que ahora el número de repostajes es limitado. Existe una cantidad de energía global en el mundo virtual desde la cual se recarga. Cada recarga solicitada repone la cantidad de fuel del vehículo desde su nivel actual hasta el nivel 100 y detrae la misma cantidad de la energía global. Cuando la energía global se agote será imposible realizar cualquier repostaje. La operación de repostaje no consume batería.
5. **PERCEPCIÓN**
   1. QUERY\_REF “” :CONVERSATION-ID @ :IN-REPLY-TO #
   2. A diferencia de la Práctica 2, en esta práctica la percepción se recibe de forma asíncrona, exclusivamente a petición de cada vehículo. En este caso recibirá un JSon indicando el estado:
      1. El nivel de batería del vehículo.
      2. Su posición en el mundo virtual (el origen de coordenadas es la esquina superior izquierda del mundo).
      3. La lectura de su sensor de radar, según el rango registrado en sus capacidades.
      4. El nivel de energía global que queda en el mundo virtual
      5. Un booleano que indica si se ha llegado al objetivo o no.
6. **CANCELACIÓN**
   1. CANCEL “”
   2. Cierra la sesión de trabajo y genera la traza de cada vehículo en un mapa privado. Cualquier vehículo puede emitir un CANCEL en cualquier momento, sin necesidad de indicar CONVERSATION-ID ni IN-REPLY-To, aunque esté averiado. Una vez recibida esta petición por el servidor, se responde de forma inmediata con AGREE y posteriormente se genera la traza de la sesión de trabajo, la cual se envía en un segundo mensaje con la performativa INFORM.

## Uso de recursos compartidos

Tanto el uso del espacio como de las reservas de energía son excluyentes, por lo que será necesario que cada equipo defina su sociedad de agentes con las características necesarias para administrar estos recursos correctamente entre todos los miembros de la sociedad.

# Seguimiento de la práctica

El profesor realizará un seguimiento semanal del estado de la implementación con cada equipo de trabajo por lo que la asistencia a las prácticas es OBLIGATORIA.

# Material necesario para la práctica

El material que sea necesario para la realización de la práctica se encuentra en la [carpeta compartida de la Práctica 3](https://drive.google.com/drive/folders/0B6yBDMinwhYEQ0VGZlA5dkh1TVk?usp=sharing).

### Ficheros relacionados

1. Guión de la práctica (este documento)
2. [Introducción a Magentix](https://docs.google.com/presentation/d/1xk_32AwNxgQIgx5l7Fc8DkaUaNzqAvbbQ8Fzr8cYj_8/edit#slide=id.g482c85ad9_0392)
3. Introducción a [miniscrum](https://docs.google.com/presentation/d/1-ApW0qx6esymUxwIUCKS_3wBJ_Kd_oMtdEBteXKdLko/edit#slide=id.p)
   1. [Hoja de seguimiento miniscrum](https://docs.google.com/spreadsheets/d/19lBvDfGRxjtWM7LTDOIN0BEa8wAsHlHa5DTvHXm6Yh8/edit#gid=0)

### Argumentos motivadores

* Preguntas abiertas
  + ¿Deben compartir los vehículos su percepción con los demás vehículos? ¿Cómo?
  + ¿Deben pasar los vehículos más de una vez por el mismo sitio? ¿Por qué?
  + ¿Cómo se reparten los distintos vehículos el objetivo?
  + Cuando uno de los vehículos descubre el objetivo y conoce sus coordenadas ¿se transforman el resto de vehículos en el GugelCar de la Práctica 2?
  + ¿Cómo meter a todos los vehículos en el área del objetivo sin chocar el que llega con los otros que ya hubiesen llegado?
  + ¿Tiene sentido que todos los vehículos vayan al mismo ritmo o que se pueda dejar “dormido” a algún vehículo temporalmente?

### Mundos posibles para conectarse

Son los mismos mundos que en la Práctica 2, pero el objetivo se encuentra en otra posición.

1. map1
2. map2
3. map3
4. map4
5. map5
6. map6
7. map7
8. map8
9. map9
10. map10

# Evaluación de la práctica

La práctica tendrá un peso de 3 puntos sobre la nota final y se evaluará de 0 a 10 puntos. Se valorarán los siguientes aspectos

1. Trabajo en equipo.
2. Modelos de los agentes, protocolos y diagramas de actividad.
3. Eficacia de la **sociedad** de agentes diseñada.

**Entrega y defensa: 18 de Enero de 2017 a las 9:30am en el aula 1.2**

# Anexo I. Diagrama de secuencia



Detalles de respuestas de error:

|  |  |
| --- | --- |
| reason1 | “BAD MAP“ |
| reason2 | "BAD PROTOCOL", "EXCEPTION CAUGHT", "ALREADY REGISTERED",  "UNREGISTERED", "BAD PERFORMATIVE OR BAD CONVERSATION" |
| reason3 | “BATTERY EXHAUSTED”, "CRASHED ON BORDER",  "CRASHED ON WALL", "CRASHED WITH AGENT" |
| reason4 | "BAD ENERGY" |

# Anexo II: Shenron el resucitador

Se ha creado un agente nuevo llamado “Shenron”, capaz de resucitar a un virtualhost que se haya quedado bloqueado, y que estará disponible para cualquier grupo en un virtualhost especial llamado “test”. Para poder interaccionar con Shenron será necesario crear un proceso nuevo en Java que requiere un AgentsConnection y un virtualhost diferente al que se usa habitualmente por cada equipo, por lo que no es posible comunicarse con Shenron desde el mismo proceso de la Práctica 3 (algo esperable si lo que se quiere es resucitar el virtualhost de la Práctica 3 que se ha quedado bloqueado).



Aunque Shenron está disponible para cualquier equipo, la comunicación con él debe hacerse mediante el “user” y “password” privados que tiene cada grupo para garantizar el funcionamiento privado de Shenron con cada equipo. De esta forma las operaciones que puede llevar a cabo Shenron son las siguientes

* Reinicio del servidor. Para ello será necesario enviar un mensaje con la performativa **REQUEST** e indicar en el contenido del mensaje el nombre de usuario y password privado de cada equipo en una cadena JSon tal y como indica el diagrama de secuencia de este anexo. Shenron reiniciará el virtualhost privado de ese equipo e informará con un mensaje de respuesta:
  + **FAILURE**. Si hubiese ocurrido algún error en la identificación del grupo o algún problema técnico inesperado al reiniciar el servidor. Sería conveniente notificarlo al profesor en un [correo electrónico](mailto:l.castillo@decsai.ugr.es).
  + **INFORM**. En caso de que el reinicio del servidor haya tenido éxito.
* Consulta del último registro de actividad del controlador privado de cada equipo. Para ello será necesario enviar un mensaje con la performativa **QUERY\_REF** e indicar en el contenido del mensaje el nombre de usuario y password privado de cada equipo en una cadena JSon tal y como indica el diagrama de secuencia de este anexo. Shenron consultará el log en disco del controlador privado de cada equipo y lo devolverá en un mensaje de respuesta:
  + **FAILURE**. Si hubiese ocurrido algún error en la identificación del grupo o algún problema técnico inesperado al consultar la información. Sería conveniente notificarlo al profesor en un [correo electrónico](mailto:l.castillo@decsai.ugr.es).
  + **INFORM**. En este caso el contenido del mensaje de respuesta contiene una cadena JSon con el último registro de actividad del agente controlador  
      
    {"date":"25/11/2016\_12:40:08", "value":{"agent":"Melania","conversation":"CONV-yxz2l5uk","content":{"status":"Waiting for SUBSCRIBE"}}}
  + Este registro contiene *fecha del registro* (comprobar que no sea demasiado antigua, lo que podría indicar que el virtualhost se hubiese quedado bloqueado), el nombre del agente controlador privado de cada equipo, la CONVERSATION-ID de la última sesión y una descripción breve del último estado en el que se encontraba el controlador cuando hizo ese apunte.